2016

Sparse Adaptive Local Machine Learning Algorithms for Sensing and Analytics

Jack Cannon
Portland State University

Hardware-friendly Sparse Coding and Machine Learning for Image Processing and Pattern Recognition Problems

Maseeh College of Engineering and Computer Science
PORTLAND STATE UNIVERSITY

teuscher.:Lab
teuscher-lab.com

Jack Cannon, Walt Woods, Jens Bürger, Christof Teuscher

Overview

Image processing is expensive. Instead of working with the original image, we can identify its most relevant components and discard the rest. This process, referred to as *sparse coding*, is also useful for identifying patterns in images. For this project, we used the MNIST database of handwritten digits to train a single-layer, hardware-friendly neural network to recognize a given digit (Fig. 1).

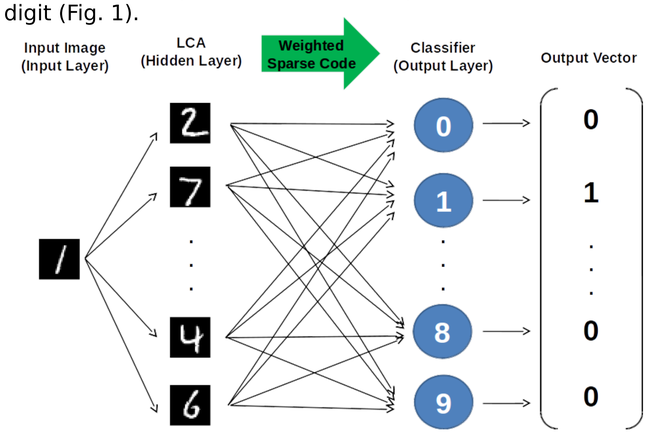


Fig. 1: An image is presented to the network. A 50-neuron hidden layer generates a sparse code and passes it to a 10-neuron classifier, which places the image into one of ten classes, each representing a digit 0-9.

Sparse Coding

Given a **dictionary** of general components, we can use a **sparse code** to select as few of them as possible to reconstruct an image of interest (Fig. 2). This reconstruction is called a **sparse representation.**

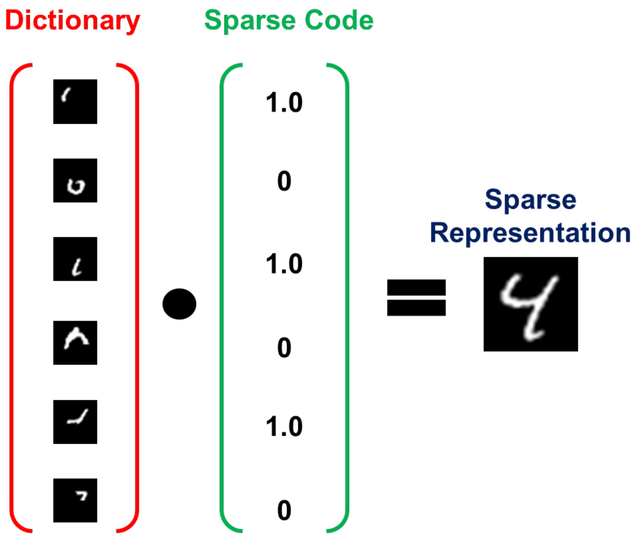


Fig. 2: A sparse representation can be thought of as the dot product of a dictionary vector and a sparse code vector.

Locally Competitive Algorithm

Rozell's (2008) Locally Competitive Algorithm (LCA) is an efficient means for utilizing highly parallel architectures to generate quality sparse codes. This is accomplished via competition among the dictionary components where those most similar to the input image prevent the others from generating a non-zero coefficient.

Dictionary Learning

There are many ways to choose a dictionary. In general, it is advantageous to use a dictionary that best represents the population of images specific to the problem at hand. To accomplish this, we started with an initial dictionary of 50 arbitrary digits and trained it to better represent the general population of MNIST digits (Fig. 3).

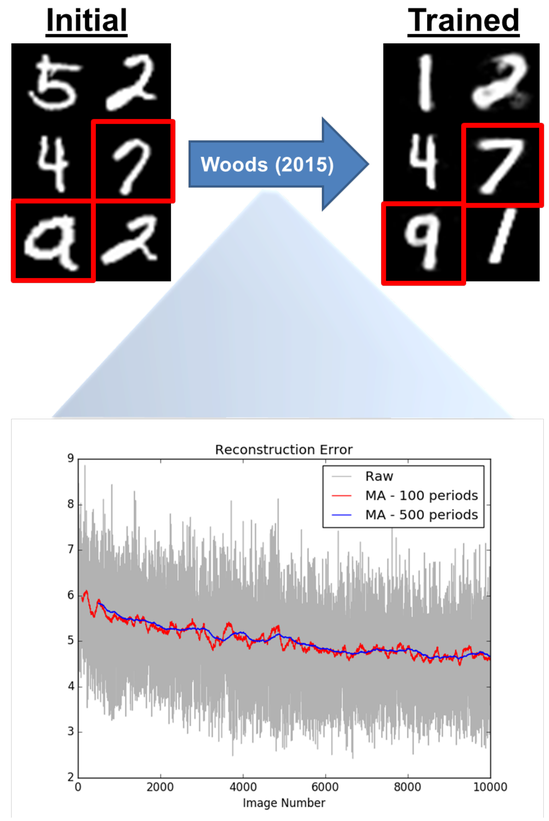


Fig. 3: A 6-image subset of the dictionary is shown before and after training. Using 10,000 images, we applied the training method from Woods (2015) to iteratively adjust the dictionary such that the reconstruction error minimizes. The highlighted digits generalized as a result of the training.

Classification

The characteristic of a sparse code that makes it useful for pattern recognition is that each non-zero coefficient contains meaningful information. This allows an image's significant features to be communicated with much less data. We used a 10-neuron sigmoid layer to take a sparse code and perform the classification step (Fig. 4).

Fig. 4: The classifier portion of the procedure, at a high level. For a given image, the LCA produces a sparse code and passes it to the sigmoid layer. Each of the ten neurons are trained to fire for their respective class of digits such that one neuron fires for each image presented to the network.

Backpropagation of Errors

The network assigns one of ten classes to each image passed to it. The difference between the predicted class and actual class (error) is used to adjust the weights applied to the sparse code such that the error minimizes (Fig. 5).

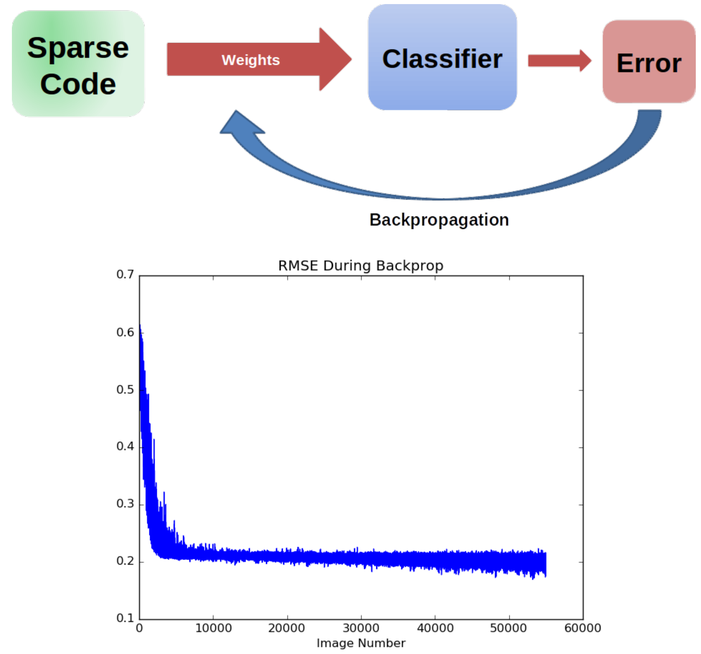


Fig 5: We used 55,000 images and trained the network via backpropagation. It is clear that only a fraction of them were needed for the error to converge.

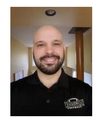
Results and Conclusion

This architecture correctly classifies MNIST digits approximately 85% of the time. There are other classifiers that achieve significantly higher levels of accuracy. However, the power of the LCA method lies in the ability to communicate the same information with less data, therefore requiring less communication bandwidth.

Acknowledgments

The authors acknowledge the support of the Semiconductor Research Corporation (SRC) Education Alliance (award # 2009-UR-2032G) and of the Maseeh College of Engineering and Computer Science (MCECS) through the Undergraduate Research and Mentoring Program (URMP).

Jack's interests are in machine learning and data science. Future projects involve exploring video processing techniques and the resulting effect on hardware. He can be reached by email at jack_cannon@live.com



W. Woods, J. Bürger, C. Teuscher, *IEEE Transactions on Nanotechnology* **14**, 945-953 (2015).
C. J. Rozell, D. H. Johnson, R. G. Baraniuk, B. A. Olshausen, Neural Computation **20**, 2526-63 (2008).